NUQ: Nonparametric Uncertainty Quantification for Deterministic Neural Networks

Nikita Kotelevskii
Skoltech
nikita.kotelevskii@skoltech.ru

Alexander Fishkov
Skoltech
alexander.fishkov@skoltech.ru

Kirill Fedyanin
Skoltech
k.fedyanin@skoltech.ru

Aleksandr Petiushko
Huawei, AIRI
petyushko@yandex.ru

Maxim Panov
Skoltech
m.panov@skoltech.ru

Abstract

This paper proposes a fast and scalable method for uncertainty quantification of machine learning models' predictions. First, we show the principled way to measure the uncertainty of predictions for a classifier based on Nadaraya-Watson's nonparametric estimate of the conditional label distribution. Importantly, the approach allows to disentangle explicitly *aleatoric* and *epistemic* uncertainties. The resulting method works directly in the feature space. However, one can apply it to any neural network by considering an embedding of the data induced by the network. We demonstrate the strong performance of the method in uncertainty estimation tasks on a variety of real-world image datasets, such as MNIST, SVHN, CIFAR-100 and several versions of ImageNet.

1 Introduction

It is crucial in many applications of modern machine learning methods to complement the prediction with some sort of a "confidence" score. In particular, deep neural network models, which usually achieve state-of-the-art results in various tasks, are notorious for providing overconfident predictions on data they did not see during training [19]. The community in recent years made tremendous efforts to develop different uncertainty estimation methods and approaches, including calibration [6], ensembling [12], Bayesian methods [5], and many others [20, 28]. Recently, a series of methods of uncertainty estimation based on the single deterministic neural network model was developed [27, 14, 26]. In practice it is usually important to distinguish two types of uncertainty: *aleatoric* and *epistemic* [3, 10]. The aleatoric uncertainty reflects the internal noise in the data due to class overlap, data markup errors, or other reasons. This type of uncertainty can not be reduced by providing more data. The epistemic uncertainty reflects the model's ignorance of data. We can reduce the uncertainty of this type once we get more data. Epistemic uncertainty, thus, may be used to identify *out-of-distribution OOD data*. If the model can quantify this type of uncertainty, it may abstain from prediction and address it to a human expert.

Summary of the contributions. We develop *a new and theoretically grounded* method of uncertainty quantification applicable to any deterministic neural network model. Our contributions:

1. We rigorously define the uncertainty of the model prediction at a particular data point. This is done by direct consideration of the probability of the wrong prediction.
2. We provide corresponding uncertainty estimate by computing the variance of the kernel estimate of conditional density with the appropriately chosen bandwidth.

35th Conference on Neural Information Processing Systems (NeurIPS 2021).

3. We implement the method in a scalable manner, which allows it to be used in the neural network's embedding space on large datasets such as ImageNet. The experimental results in misclassification detection and OOD detection tasks show the significant potential of the proposed approach.

2 Nonparametric Uncertainty Quantification

2.1 Estimation under Covariate Shift

Consider a binary classification setup $(X, Y) \in \mathbb{R}^d \times \{0, 1\}$ with $(X, Y) \sim \mathbb{P}_{\mathrm{tr}}$. We assume that we observe the dataset $\mathcal{D} = \big\{(X_i, Y_i)\big\}_{i=1}^n$ of *i.i.d.* points from $\mathbb{P}_{\mathrm{tr}}$. The classical problem is to find a rule $\hat{g}$ based on the dataset $\mathcal{D}$ which approximates the optimal one: $g^* = \arg\min_g \mathbb{P}(g(X) \neq Y)$. Here $g \colon \mathbb{R}^d \to \{0, 1\}$ is any classifier and the probability of wrong classification $\mathcal{R}_g = \mathbb{P}(g(X) \neq Y)$ is usually called *risk*. The rule g^* is given by the *Bayes optimal classifier*:

$$g^*(x) = \begin{cases} 1, & \eta(x) \geq \frac{1}{2}, \\ 0, & \eta(x) < \frac{1}{2}, \end{cases}$$

where $\eta(x) = p(Y = 1 \mid X = x)$ under the distribution $\mathbb{P}$.

We consider a situation when the distribution of the test samples $\mathbb{P}_{\mathrm{test}}$ is different from the one for the training dataset $\mathbb{P}_{\mathrm{tr}}$. The rule g^* obtained for $\mathbb{P} = \mathbb{P}_{\mathrm{tr}}$ might no longer be optimal if we minimize the error on the test data $\mathbb{P}_{\mathrm{test}}(g(X) \neq Y)$.

For a meaningful estimation problem, some additional assumptions are needed. First, we assume that the distribution $\mathbb{P}_{\mathrm{test}}$ is unknown at the model construction moment, only the dataset $\mathcal{D}$ is available. Also, we assume that the distribution $p(y \mid x)$ is the same under both $\mathbb{P}_{\mathrm{tr}}$ and $\mathbb{P}_{\mathrm{test}}$, which means that: 1) All the difference between $\mathbb{P}_{\mathrm{tr}}$ and $\mathbb{P}_{\mathrm{test}}$ is due to the difference between marginal distributions of X: $p_{\mathrm{train}}(X)$ and $p_{\mathrm{test}}(X)$. 2) The Bayes rule is still valid, i.e., optimal even under $\mathbb{P}_{\mathrm{test}}$. However, while the rule g^* is still optimal, its approximation $\hat{g}$ might be arbitrary bad under the covariate shift.

2.2 Problem Statement

Consider a classification rule $\hat{g}(x) = \hat{g}_{\mathcal{D}}(x)$ on the dataset $\mathcal{D}$. Define pointwise risk of estimation:

$$\mathcal{R}(x) = \mathbb{P}(\hat{g}(X) \neq Y \mid X = x),$$

where $\mathbb{P}(\hat{g}(X) \neq Y \mid X = x) \equiv \mathbb{P}_{\mathrm{tr}}(\hat{g}(X) \neq Y \mid X = x) \equiv \mathbb{P}_{\mathrm{test}}(\hat{g}(X) \neq Y \mid X = x)$ under the assumptions above. The value $\mathcal{R}(x)$ is independent of covariate distribution $p_{\mathrm{test}}(X)$ and allows to define a meaningful target of estimation which is based solely on the quantities known for the training distribution.

Let us note that the total risk value $\mathcal{R}(x)$ admits the following decomposition: $\mathcal{R}(x) = \tilde{\mathcal{R}}(x) + \mathcal{R}^*(x)$, where $\mathcal{R}^*(x) = \mathbb{P}(g^*(X) \neq Y \mid X = x)$ is Bayes risk and $\tilde{\mathcal{R}}(x) = \mathbb{P}(\hat{g}(X) \neq Y \mid X = x) - \mathbb{P}(g^*(X) \neq Y \mid X = x)$ is an excess risk. Here $\mathcal{R}^*(x)$ corresponds to aleatoric uncertainty as it completely depends on the data distribution. Excess risk $\tilde{\mathcal{R}}(x)$ directly measures imperfectness of the model $\hat{g}$ and thus can be seen as a measure of epistemic uncertainty.

To proceed, we first assume that the classifier $\hat{g}$ has the form of optimal Bayesian classifier with respect to the density $\hat{\eta}(x) = \hat{p}(Y = 1 \mid X = x)$, which is an estimate of the conditional density $\eta(x)$. We can efficiently bound the excess risk via the following classical inequality [4]:

$$\tilde{\mathcal{R}}(x) = \mathbb{P}(\hat{g}(X) \neq Y \mid X = x) - \mathbb{P}(g^*(X) \neq Y \mid X = x) \leq 2|\hat{\eta}(x) - \eta(x)|.$$

It allows us to obtain an upper bound for the risk: $\mathcal{R}(x) \leq \mathcal{L}(x) = \mathcal{R}^*(x) + 2|\hat{\eta}(x) - \eta(x)|$, where $\mathcal{R}^*(x) = \min\{\eta(x), 1 - \eta(x)\}$ is just the Bayes risk.

2.3 Nonparametric Uncertainty Quantification

2.3.1 Kernel Density Estimate and Its Asymptotic Distribution

For the approach above we need to consider some particular type of estimator for $\hat{g}$. We consider kernel-based estimator of the conditional density because of its asymptotic properties. For a class label c, the conditional probability estimate can be expressed as:

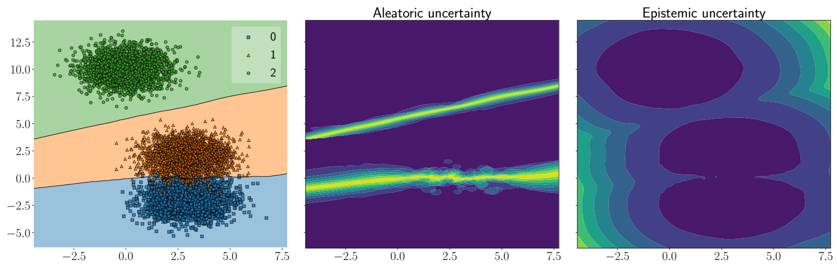

Figure 1: Left plot shows the data and the result of the classification by the Bayes classifier based on the nonparametric estimate of conditional density. The next two plots show different types of uncertainties: "aleatoric" and "epistemic". The lighter color, the higher uncertainty. We see that the former does not increase as we go away from training data, while the latter does.

$$\hat{p}(Y = c \mid X = x) = \frac{\sum_{i=1}^{N} K_h(x_i - x)[y_i = c]}{\sum_{j=1}^{N} K_h(x_j - x)}. \tag{1}$$

We consider 1D kernel function $K\colon \mathbb{R} \to \mathbb{R}_+$ and construct the resulting kernel in $\mathbb{R}^d$ of the form: $K_h(x - y) = \prod_{j=1}^{d} K\left(\frac{x^j - y^j}{h}\right)$. It could be shown, that the difference $\hat{\eta}(x) - \eta(x)$ converges in distribution as follows (see, e.g. [22]):

$$\hat{\eta}(x) - \eta(x) \to \mathcal{N}\left(0, \frac{1}{Nh^d}\frac{\sigma^2(x)}{p(x)}\left[\int [K(u)]^2 du\right]^d\right), \tag{2}$$

where n is the number of data points in the training set, $K(\cdot)$ is the kernel used for kernel density estimate (KDE), h is the bandwidth of the kernel; d is the dimensionality of the problem and $\sigma^2(x)$ is the standard deviation of the data label at point x. Let us denote the variance term in (2) by $\tau^2(x)$.

In this work, we suggest to consider the following measure of the total uncertainty:

$$\mathbf{U}_t(x) = \min\bigl\{\eta(x), 1 - \eta(x)\bigr\} + 2\sqrt{\frac{2}{\pi}}\tau(x),$$

which is obtained by considering an asymptotic approximation of $\mathbb{E}_{\mathcal{D}}\mathcal{L}(x)$.

To obtain the practical estimate, the integral $\int [K(u)]^2 du$ can be computed in the closed form for various standard kernels, see Supplementary Material, Table 3. Second, we approximate the marginal density of objects $p(x)$. The density can be again obtained via KDE (but one can choose another estimation [17]): $\hat{p}(x) = \frac{1}{Nh^d}\sum_{i=1}^{N} K_h(x - x_i)$. The only thing left is the variance which can be estimated as $\hat{\sigma}^2(x) = \hat{\sigma}^2(y|x) = \hat{\eta}(x)(1 - \hat{\eta}(x))$. We refer our readers to Supplementary Material to for more details on computation of different uncertainties.

3 Experiments

3.1 Toy Example

We start this section with the application of the proposed *Nonparametric Uncertainty Quantification (NUQ)* method to a toy example. As a dataset, we use a 2-dimensional mixture of three Gaussians with centers at points [3, -2], [3, 2], [0, 10], and variance equal to 1. Each Gaussian is treated as a separate class (see Figure 1, the leftmost panel).

We consider the Bayes classifier based on the nonparametric estimate of the conditional density (1) and compute aleatoric and epistemic uncertainty values according to equations (3). Bandwidth was selected according to Improved Sheather–Jones ("ISJ") rule [1] independently for each data dimension. Classification results and uncertainties for this toy problem are presented in Figure 1. The first plot shows the raw data and the result of the classification by the Bayes rule. Two other plots present aleatoric and epistemic uncertainty estimates obtained. The uncertainty measures show the desired behavior: aleatoric uncertainty is large in-between the classes, while epistemic uncertainty increases with the increase of the distance to the training data.

OOD dataset	MaxProb*	Entropy*	Dropout	Ensemble	TTA	DDU*	NUQ*
SVHN	79.7±1.3	81.1±1.6	77.6±2.5	82.9±0.9	81.6±1.2	**89.6±1.6**	**89.7±1.6**
LSUN	81.5±2.0	83.0±2.1	76.8±5.1	86.5±0.8	85.0±2.7	**92.1±0.6**	**92.3±0.6**
Smooth	76.6±3.5	77.8±5.2	63.3±3.8	83.7±1.2	73.2±10.8	**97.1±3.1**	**96.8±3.8**

Table 1: OOD detection for CIFAR-100 in-distribution dataset with ResNet-50 neural network. The top two results are shown in bold. Evaluation is done for three models trained with different seeds to estimate the standard deviation. Methods requiring a single pass over the data to compute uncertainty estimates are marked with *.

OOD dataset	MaxProb*	Entropy*	TTA	Ensemble	DDU*	NUQ*
ImageNet-R	80.4	83.6	85.8	84.4	80.1	**99.5**
ImageNet-O	28.2	29.1	30.5	51.9	74.1	**82.4**

Table 2: ROC-AUC score for ImageNet out-of-distribution detection tasks for different methods. Methods requiring a single pass over the data to compute uncertainty estimates are marked with *.

3.2 Image Classification Datasets

In this section, we consider a series of experiments on image datasets. In contrast to the toy example above, we should first train a model and then apply NUQ to its predictive features. We emphasise, that NUQ is the postprocessing method, which is fitted to the embeddings obtained from a given model. In what follows, we call this model a "base model". In the experiments of this section, we use logits as extracted features, if not explicitly stated otherwise. However, other options are also possible; see Supplementary Material, Section A.5. In experiments below, we compare our method to several baselines. See Supplementary Material to find a brief description of them.

CIFAR-100. To reinforce our results on simpler datasets, we further conduct experiments on more challenging CIFAR-100 [11]. We want our model to detect the unconventional samples, and thus we treat the out-of-distribution detection as a binary classification task (OOD/not-OOD) by uncertainty score, and we report the ROC-AUC for that task. Following the setup from the recent works [27, 26, 23], we use SVHN, LSUN [29] and Smooth [7] datasets as OOD datasets.

We trained the ResNet-50 model from scratch on CIFAR-100. For our method and DDU, we use training with spectral normalization [16] to ensure the bi-Lipshitz constraint for mappings at each layer. In this experiment, NUQ was applied to the features from the penultimate layer, and the density estimate is given by GMM. See the results for other choices of hyperparameters in the Supplementary Material, Section A.5.

The results are presented in Table 1. The ensemble has a strong performance, which is expected. The TTA performs reasonably well with the quality close to the one of the ensemble. We can clearly see that NUQ and DDU show close results while outperforming the competitors with a significant margin.

ImageNet. To evaluate the method's applicability to the large-scale data, we have applied our approach to the ImageNet [2] dataset. As OOD data we used the ImageNet-O [9] and ImageNet-R[8] datasets. ImageNet-O consists of images from classes that are not found in the standard ImageNet-1k dataset. ImageNet-R contains different artistic renditions of ImageNet classes. It turned out that in these experiments, NUQ beats all the competitors with a large margin; see Table 2.

4 Conclusions

In this work, we propose NUQ, a new principled uncertainty estimation method that applies to a wide range of neural network models. It does not require retraining the model and acts as a postprocessing step working in the embedding space induced by the neural network. NUQ significantly outperforms the competing approaches with only recently proposed DDU method [17] showing comparable results. Importantly, in the most practical example of OOD detection for ImageNet data, NUQ shows the best results with a significant margin.

Bibliography

[1] Zdravko I Botev, Joseph F Grotowski, Dirk P Kroese, et al. Kernel density estimation via diffusion. *The annals of Statistics*, 38(5):2916–2957, 2010.

[2] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[3] Armen Der Kiureghian and Ove Ditlevsen. Aleatory or epistemic? does it matter? *Structural safety*, 31(2):105–112, 2009.

[4] Luc Devroye, László Györfi, and Gábor Lugosi. *A probabilistic theory of pattern recognition*, volume 31. Springer Science & Business Media, 2013.

[5] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In Maria-Florina Balcan and Kilian Q. Weinberger, editors, *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, volume 48 of *JMLR Workshop and Conference Proceedings*, pages 1050–1059. JMLR.org, 2016.

[6] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In *International Conference on Machine Learning*, pages 1321–1330. PMLR, 2017.

[7] Matthias Hein, Maksym Andriushchenko, and Julian Bitterwolf. Why relu networks yield high-confidence predictions far away from the training data and how to mitigate the problem. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 41–50, 2019.

[8] Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu, Samyak Parajuli, Mike Guo, Dawn Song, Jacob Steinhardt, and Justin Gilmer. The many faces of robustness: A critical analysis of out-of-distribution generalization. *ICCV*, 2021.

[9] Dan Hendrycks, Kevin Zhao, Steven Basart, Jacob Steinhardt, and Dawn Song. Natural adversarial examples. *CVPR*, 2021.

[10] Alex Kendall and Yarin Gal. What uncertainties do we need in bayesian deep learning for computer vision? In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5574–5584, 2017.

[11] Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, 2009.

[12] Balaji Lakshminarayanan, A. Pritzel, and C. Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *NIPS*, 2017.

[13] Yann LeCun, Corinna Cortes, and CJ Burges. Mnist handwritten digit database. *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, 2, 2010.

[14] Jeremiah Z. Liu, Zi Lin, Shreyas Padhy, Dustin Tran, Tania Bedrax-Weiss, and Balaji Lakshminarayanan. Simple and principled uncertainty estimation with deterministic deep learning via distance awareness. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

[15] Yu A Malkov and Dmitry A Yashunin. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *IEEE transactions on pattern analysis and machine intelligence*, 42(4):824–836, 2018.

[16] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018.

[17] Jishnu Mukhoti, Andreas Kirsch, Joost van Amersfoort, Philip H. S. Torr, and Yarin Gal. Deterministic neural networks with appropriate inductive biases capture epistemic and aleatoric uncertainty. *CoRR*, abs/2102.11582, 2021.

[18] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2011.

[19] Anh M Nguyen, J. Yosinski, and J. Clune. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 427–436, 2015.

[20] Yaniv Ovadia, E. Fertig, J. Ren, Zachary Nado, D. Sculley, S. Nowozin, Joshua V. Dillon, Balaji Lakshminarayanan, and Jasper Snoek. Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift. In *NeurIPS*, 2019.

[21] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

[22] James L. Powell. Notes on nonparametric regression estimation. *Manuscript*, 2010.

[23] Chandramouli Shama Sastry and Sageev Oore. Detecting out-of-distribution examples with gram matrices. In *ICML*, 2020.

[24] David W Scott. On optimal and data-based histograms. *Biometrika*, 66(3):605–610, 1979.

[25] Bernard W Silverman. *Density estimation for statistics and data analysis*. Routledge, 2018.

[26] Joost van Amersfoort, Lewis Smith, Andrew Jesson, Oscar Key, and Yarin Gal. Improving deterministic uncertainty estimation in deep learning for classification and regression. *CoRR*, abs/2102.11409, 2021.

[27] Joost Van Amersfoort, Lewis Smith, Yee Whye Teh, and Yarin Gal. Uncertainty estimation using a single deep deterministic neural network. In *International Conference on Machine Learning*, pages 9690–9700. PMLR, 2020.

[28] G. Wang, Wenqi Li, M. Aertsen, J. Deprest, S. Ourselin, and Tom Kamiel Magda Vercauteren. Aleatoric uncertainty estimation with test-time augmentation for medical image segmentation with convolutional neural networks. *Neurocomputing*, 335:34 – 45, 2019.

[29] Fisher Yu, Yinda Zhang, Shuran Song, Ari Seff, and Jianxiong Xiao. LSUN: construction of a large-scale image dataset using deep learning with humans in the loop. *CoRR*, abs/1506.03365, 2015.

A Appendix

A.1 How to Choose Bandwidth Properly?

The choice of the optimal bandwidth parameter is well-developed in the theory of kernel density estimation. For example, one can base on asymptotically optimal values and select the bandwidth accordingly as in Silverman's [25] or Scott's [24] rules. However, such estimates are usually very crude in practice.

A.2 How to Compute Kernel Estimate when N is Large?

Our nonparametric method involves a sum over the whole available data to compute the estimates. This could be intractable in practice when we are working with large datasets. However, the typical kernel K_h quickly approaches zero with the increase of the norm of the argument: $\|x - x_i\|$. Thus, we can use an approximation of kernel estimates: instead of the sum over all elements in the dataset, we consider the contribution of only several nearest neighbors. It requires a fast algorithm for finding the nearest neighbors. For this purpose, we use the approach of [15] based on Hierarchical Navigable Small World graphs (HNSW). It provides a fast, scalable, and easy-to-use solution to the computation of nearest neighbors.

A.3 Multiclass Generalization for Uncertainties

In this section we show, how our method can be generalized from binary classification to multiclass problems. Consider data pairs $(X, Y) \sim \mathbb{P}$. Now, $X \in \mathbb{R}^d$ and $Y \in 1, \dots, C$, where C is the number of classes. We also denote $\eta_c(x) = \mathbb{P}(Y = c \mid X = x)$.

Let us start with the Bayes risk:

$$\begin{aligned} &\mathbb{P}(Y \neq g^*(X) \mid X = x) = 1 - \mathbb{P}(Y = g^*(X) \mid X = x) \\ = \; &1 - \max_c \eta_c(x) = \min_c \bigl\{1 - \eta_c(x)\bigr\}, \end{aligned}$$

where $g^*(x) := \arg\max_c \eta_c(x)$ is Bayes optimal classifier.

Let us further move to the excess risk. Denote by $\hat{\eta}_c(x)$ density, we approximate. Analogously, $g(x) := \arg\max_c \hat{\eta}_c(x)$

$$\begin{aligned} &\mathbb{P}(Y \neq g(X) \mid X = x) - \mathbb{P}(Y \neq g^*(X) \mid X = x) = \eta_{g^*(x)}(x) - \eta_{g(x)}(x) \\ = \; &\eta_{g^*(x)}(x) - \hat{\eta}_{g^*(x)}(x) + \hat{\eta}_{g^*(x)}(x) - \hat{\eta}_{g(x)}(x) + \hat{\eta}_{g(x)}(x) - \eta_{g(x)}(x) \\ \leq \; &\bigl|\eta_{g^*(x)}(x) - \hat{\eta}_{g^*(x)}(x)\bigr| + \bigl|\eta_{g(x)}(x) - \hat{\eta}_{g(x)}(x)\bigr|, \end{aligned}$$

where we used the fact that $\hat{\eta}_{g^*(x)}(x) - \hat{\eta}_{g(x)}(x) \leq 0$ for any x.

The expectation of the right hand can be upper bounded by $2\sqrt{\frac{2}{\pi}}\tau(x)$, where $\tau(x)$ is defined below.

Total uncertainty for multiclass problem is thus

$$\mathbf{U}_t(x) = \min_c \bigl\{1 - \eta_c(x)\bigr\} + 2\sqrt{\frac{2}{\pi}}\tau(x),$$

where

$$\tau^2(x) = \frac{1}{Nh^d} \frac{\max_c \bigl\{\sigma_c^2(x)\bigr\}}{p(x)} \int \bigl[K(u)\bigr]^2 du$$

and $\sigma_c^2(x) = \eta_c(x)\bigl(1 - \eta_c(x)\bigr)$.

A.4 Architectures

A.4.1 Base Model

For CIFAR-100 and ImageNet-like datasets, we are using ResNet50 with or without spectral normalization [16]. For the spectral normalization, we use 3 iterations of the power method. We use a ResNet50 architecture with implementation from PyTorch [21]. This architecture was implemented for the ImageNet dataset; thus, for the CIFAR-100, we had to adapt it. We changed the first convolutional layer and used kernel size 3x3 with stride 1 and padding 1 (instead of kernel size 7x7 with stride 2 and padding 3). For CIFAR-100, we train the model for 200 epochs with an SGD optimizer, starting with a learning rate of 0.1 and decaying it 5 times on 60, 120, and 160 epoch. For ImageNet, we train the model for 90 epochs with an SGD optimizer learning rate decaying 10 times every 30 epochs.

For MNIST, we train a small convolutional neural network with three convolution layers with padding of 1 and kernel size of 3. Each of these layers is followed by a batch normalization layer. Finally, it has a linear layer with Softmax activation. This network achieves an accuracy of 0.99 on the holdout set.

We refer readers to our code for more specific details.

A.4.2 Ensemble

For ensemble with use a combination of 5 base models, trained with different random seeds.

A.4.3 Test-Time Augmentation (TTA)

For TTA, we use a base model with applying a transformation on the inference stage. Images of CIFAR-100 are randomly cropped with padding 4, randomly horizontally flipped, and randomly rotated up to 15 degrees. ImageNet is randomly cropped from 256 to 224, randomly horizontally flipped, and the color was jittered (0.02).

A.4.4 Spectrally Normalized Models

For both DDU and NUQ, we need spectral normalized models to extract features. We're wrapping each convolutional and linear layer with spectral normalization (PyTorch implementation). We used 3 iterations of the power method in our experiments.

A.5 Ablation Study on CIFAR-100

A.5.1 Choice of Kernel for Uncertainty Quantification

In this section, we study the choice of a kernel for uncertainty quantification.

We consider the following choices:

Kernel name	**Formula** $K(u)$	**Integral** $\int K(u)^2 du$
Gaussian (RBF)	$\frac{1}{\sqrt{2\pi}} \exp\left\{-u^2\right\}$	$\frac{1}{2\sqrt{\pi}}$
Sigmoid	$\frac{2}{\pi} \frac{1}{\exp\{-u\}+\exp\{u\}}$	$\frac{2}{\pi^2}$
Logistic	$\frac{1}{\exp\{-u\}+2+\exp\{u\}}$	$\frac{1}{6}$

Table 3: Different types of kernels $K(u)$ considered and corresponding values of the integral $\int K(u)^2 du$.

We need a probability density estimation for our method, and there are different options: we consider kernel method with RBF kernel and logistic kernel and Gaussian mixtures models. There is also a question about which embeddings to use - the DDU paper proposes to take the features from the second last layer; we believe the logits from the last layer are a reasonable choice as well. To validate the options, we conducted some ablation study on out-of-distribution detection for the CIFAR-100 dataset, similar to the main experiment.

First, we compare the DDU and NUQ on embeddings from the pre-last and last layer (Table 4) on SVHN, LSUN, and Smooth datasets. Secondly, we compare the NUQ method on RBF, logistic kernel, and GMM for both last and penultimate layer embeddings(Table 5). As we can see from the tables, the optimal is the option with GMM density on the penultimate layer.

Kernel-based methods rely on the "reasonable" geometry of the embedding space, meaning that embeddings of similar images should not be too far and different images should not collapse into a single point. Our motivation to use spectral normalization during training is to make the embedding space more smooth with respect to input images. We have conducted an extra ablation study, comparing the result for feature extractors with and without spectral normalization, see Table 6. The

	DDU, features	DDU, logits	NUQ, features	NUQ, logits
SVHN	89.6±1.6	88.2±0.6	89.7±1.6	88.2±0.6
LSUN	92.1±0.6	90.9±0.4	92.3±0.6	90.9±0.4
Smooth	97.1±3.1	96.3±4.1	96.8±3.8	96.2±4.1

Table 4: Comparison of DDU and NUQ predictions on different type of embeddings - logits (last layer) and features (second last layer).

	RBF, f	RBF, l	Logistic, f	Logistic, l	GMM, f	GMM, l
SVHN	84.4±3.2	84.7±3.1	84.8±2.9	86.7±2.6	89.7±1.6	88.2±0.6
LSUN	88.2±1.0	88.1±0.8	88.5±4.0	90.3±1.0	92.3±0.6	90.9±0.4
Smooth	85.5±6.8	87.7±9.4	86.2±8.2	90.8±7.8	96.8±3.8	96.2±4.1

Table 5: Probability density methods comparison – radial basis function kernel (RBF), logistic kernel, gaussian mixture models (GMM). 'f' (Features) marks models, built on embeddings from a second last layer and 'l' (logits) is for the ones built on embeddings from a last layer.

results confirm our hypothesis, as the spectral-normalized version performs better, though the NUQ beats the baseline even without applying the modification to the ResNet training. We also show here that entropy performs better than maximum probability as an uncertainty measure.

OOD dataset	MaxProb	Entropy	DDU	DDU (spectral)	NUQ	NUQ (spectral)
SVHN	79.7±1.3	81.1±1.6	88.7±4.3	89.6±1.6	86.8±1.2	89.7±1.6
LSUN	81.5±2.0	83.0±2.1	91.3±0.9	92.1±0.6	91.2±1.1	92.3±0.6
Smooth	76.6±3.5	77.8±5.2	95.7±1.2	97.1±3.1	95.5±1.3	96.8±3.8

Table 6: Comparing the influence of spectral normalization on the model performance for OOD detection, ROC-AUC.

A.6 Estimates of Total, Aleatoric and Epistemic Uncertainty

Let us denote by $\tau(x)$ the standard deviation of a Gaussian from equation (2):

$$\tau^2(x) = \frac{1}{Nh^d}\frac{\sigma^2(x)}{p(x)}\biggl[\int [K(u)]^2 du\biggr]^d.$$

In this work, we suggest to consider the following measure of the total uncertainty:

$$\mathbf{U}_t(x) = \min\bigl\{\eta(x), 1 - \eta(x)\bigr\} + 2\sqrt{\frac{2}{\pi}}\tau(x),$$

which is obtained by considering an asymptotic approximation of

$$\mathbb{E}_{\mathcal{D}}\mathcal{L}(x) = \min\bigl\{\eta(x), 1 - \eta(x)\bigr\} + 2\mathbb{E}_{\mathcal{D}}\bigl|\hat{\eta}(x) - \eta(x)\bigr|$$

in a view of (2) and the fact, that $\mathbb{E}|\xi| = \mathrm{std}(\xi)\sqrt{\frac{2}{\pi}}$ for the zero-mean normal variable ξ. The resulting estimate upper bounds the average error of estimation at point x and thus indeed can be used as the measure of total uncertainty.

We also can write the corresponding measures of aleatoric and epistemic uncertainties:

$$\mathbf{U}_a(x) = \min\bigl\{\eta(x), 1 - \eta(x)\bigr\}, \qquad \mathbf{U}_e(x) = 2\sqrt{\frac{2}{\pi}}\tau(x). \tag{3}$$

Finally, the data-driven uncertainty estimates $\hat{\mathbf{U}}_t(x)$ and $\hat{\mathbf{U}}_e(x)$ can be obtained via plug-in using estimates $\hat{\eta}(x)$, $\hat{\sigma}(x)$, $\hat{p}(x)$ and, consequently, $\hat{\tau}^2(x) = \frac{1}{Nh^d}\frac{\hat{\sigma}^2(x)}{\hat{p}(x)}\biggl[\int\bigl[K(u)\bigr]^2 du\biggr]^d$.

The generalization of the considered uncertainty measures to the case of multiple classes results in the total uncertainty given by

$$\mathbf{U}_t(x) = \min_c \bigl\{1 - \eta_c(x)\bigr\} + 2\sqrt{\frac{2}{\pi}}\tau(x),$$

where $\tau^2(x) = \frac{1}{Nh^d} \frac{\max_c \bigl\{\sigma_c^2(x)\bigr\}}{p(x)} \left[\int \bigl[K(u)\bigr]^2 du\right]^d$ and $\sigma_c^2(x) = \eta_c(x)\bigl(1 - \eta_c(x)\bigr)$. The derivation of these formulas can be found in Supplementary Material, Section A.3. We note that the resulting formula for aleatoric uncertainty $\mathbf{U}_a(x) = \min_c \bigl\{1 - \eta_c(x)\bigr\}$ coincides with classical maximum probability (MaxProb) uncertainty measure.

The only remaining unspecified ingredient of the procedure is the choice of bandwidth h for KDE (see Appendix).

In this work, we consider the choice of bandwidth based on the Improved Sheather–Jones algorithm [1]. We assume that the bandwidth optimal for the primary problem (density estimation) is also helpful for OOD detection. It is not necessarily so in practice. Thus, it might be beneficial to tune the bandwidth to optimize the quality of OOD detection if some set of OOD points is available at the training time. However, we find that considered estimates perform fairly well in practice, see the experimental evaluation in Section 3.

A.7 Baseline description

We compare popular measures of uncertainty which do not require significant modifications to model architectures and training procedures. More specifically, we consider:

1. Maximum probability (MaxProb): $1 - \max_c p(y = c \mid x)$;
2. Entropy: $-\sum_{c=1}^{C} p(y = c \mid x) \log p(y = c \mid x)$;
3. Monte-Carlo dropout [5];
4. Ensemble of models trained with different random seeds;
5. Test-Time Augmentation (TTA) – augmentation, applied to data at inference time;
6. DDU [17] involves Gaussian Mixture Model (GMM)-like approximation of extracted features to predict uncertainties.

For Monte-Carlo dropout, Ensembles, and TTA, we first compute average vectors of predictions and then compute its entropy (as we noticed) among MaxProb, Standard deviation, and BALD entropy provides the best ROC-AUC results). More details can be found in Supplementary Material, Section A.4.

A.8 Additional experiments

A.8.1 Rotated MNIST

The second example is misclassification detection on MNIST [13]. We train a small convolutional neural network with three convolution layers, see Supplementary Material, Section **??**. This is the base model we use to obtain logits for the input objects. We consider a particular instance of distribution shift for evaluation by using a test set of MNIST images rotated at a random angle in the range from 45 to 90 degrees. This set contains 10000 images. The range of angles reassures that the data does not look like the original MNIST data, though many resulting pictures can still remind the ones from training.

In this experiment, we consider MaxProb and Entropy-based uncertainty estimates of the base model (using base model predictions, not NUQ) and compare them with NUQ-based estimate of total uncertainty $\hat{\mathbf{U}}_t(x)$. To evaluate the quality of the uncertainty estimates, we sort the objects from the test dataset in order of ascending uncertainties. Then we obtain the model's predictions and plot how accuracy changes with the number of objects taken into consideration; see Figure 2. The valid uncertainty estimation method is expected to produce the plot with accuracy decreasing when

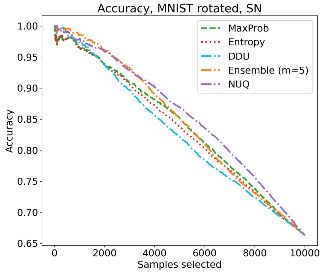

Figure 2: Accuracy for images sorted by uncertainty on rotated MNIST.

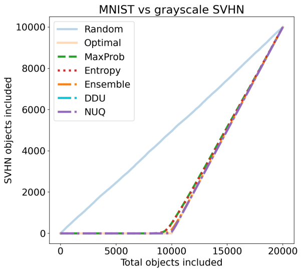

Figure 3: Share of SVHN images included into consideration vs unrotated MNIST.

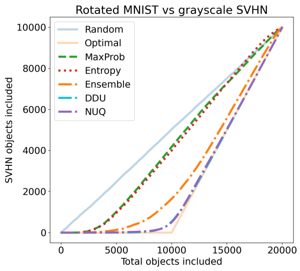

Figure 4: Share of SVHN images included into consideration vs rotated MNIST.

more samples are taken into account. Moreover, the higher is the plot, the better is the quality of the corresponding uncertainty estimate. We see that the plots for all the considered methods show the expected trend, while uncertainties obtained by NUQ are more reliable.

A.8.2 MNIST vs. SVHN

To make the problem more challenging, we consider the SVHN dataset [18], convert it to grayscale, and resize it to the shape of 28 x 28. The size of this additional SVHN-based dataset is again 10000. We take the base model trained on MNIST from the previous section and consider the problem of OOD detection with SVHN being the OOD dataset.

As in-distribution, we first consider the test set of 10000 MNIST images. We again compute uncertainties for each object of this concatenated dataset (10000 of MNIST and 10000 of SVHN) and sort them by their uncertainties in ascending order. For NUQ we use total uncertainty $\hat{\mathbf{U}}_t(x)$ in this experiment. In Figure 3 we plot the share of objects included from the SVHN dataset. It is clearly seen that NUQ assigns higher uncertainties to objects from SVHN. In fact, NUQ almost perfectly separates MNIST from SVHN (optimal result is also depicted on the plot). Although NUQ is the leader in this task, competitors show good performance, and we move on to make the problem more challenging.

We consider the problem of separation between rotated MNIST (see Section A.8.1) and SVHN. We expect that it is harder to distinguish between them as rotated MNIST images differ from those used to train the network. However, Figure 4 shows that NUQ still does a very good job and allows for almost perfect separation. Interestingly, other methods completely fail and perform no better than random baseline.